DOCUMENT RESUME

ED 427 684 IR 019 235

AUTHOR Badue, Claudine; Vaz, Wesley; Albuquerque, Eduardo
TITLE Using an Automatic Retrieval System in the Web To Assist Co-operative Learning.
PUB DATE 1998-11-00
NOTE 7p.; In: WebNet 98 World Conference of the WWW, Internet, and Intranet Proceedings (3rd, Orlando, FL, November 7-12, 1998); see IR 019 231.
PUB TYPE Reports - Descriptive (141) -- Speeches/Meeting Papers (150)
EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Automation; *Computer System Design; Computer Uses in Education; Cooperative Learning; Foreign Countries; Higher Education; Indexing; *Information Retrieval; Information Seeking; *Information Systems; Keywords; Relevance (Information Retrieval); Semantics; User Needs (Information); *World Wide Web
IDENTIFIERS Brazil; Filters; *Links (Indexing); *Search Engines

ABSTRACT

This paper presents an information agent and latent semantic-based indexing architecture to retrieve documents on the Internet. The system optimizes the search for documents in the Internet by automatically retrieving relevant links. The information used for the search can be obtained, for instance, from Internet browser caches and from grades of relevance manually informed. To leverage the scope of retrieved documents, the system makes use of existing indexing mechanisms. Returned documents are then filtered using Latent Semantic Indexing (LSI). In a co-operative environment, the proposed architecture provides for sharing of documents and grades among the group. The architecture has been used in a cooperative learning environment, where students share their browser caches and retrieved documents. The paper focuses on the architecture of the information agent; the context reconnaissance, search, and filter modules are described. Scenarios of usage and system performance are also addressed. Three figures present the architecture of the agent, the keywords generation process, and a mathematical representation of the decomposition into singular value used to implement LSI. (Author/AEF)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED 427 684

Using an Automatic Retrieval System in the Web to Assist Co-operative Learning

Claudine Badue
claudine@genetic.com.br

Wesley Vaz
wvaz@uol.com.br

Eduardo Albuquerque
eduardo@inf.ufg.br

Universidade Federal de Goiás
Instituto de Informática
Campus Samambaia - IMF I
Goiânia-GO BRAZIL

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
☐ This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
G.H. Marks
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC).

Abstract:

We present an information agent and latent semantic based indexing architecture to retrieve documents on the internet. This system optimises the search for documents in the internet by automatically retrieving relevant links. The information used for the search can be obtained, for instance, from Internet browser caches and from grades of relevance manually informed. To leverage the scope of retrieved documents, the system makes use of existing indexing mechanisms. Returned documents are then filtered using Latent Semantic Indexing (LSI). In a co-operative environment, the proposed architecture provides for sharing of documents and grades among the group. The architecture has been used in a "Co-operative Learning Environment" where students share their browser caches and retrieved documents

1 Introduction

The huge amount of information available in the Internet allows research on virtually any subject. However, this wealth of data makes it almost impossible to retrieve relevant documents for, say, a school project. Therefore, we need automatic methods to retrieve information in an easy and significant way. If we can make the process transparent its usefulness would be even bigger.

Some of the common solutions available today include search using indexing servers. The problem with this approach is that the user must explicitly select keywords, activate the search mechanism, wait for the response and identify the relevant documents returned. All steps require user input. We propose an architecture that makes use of existing indexing mechanisms but automatically filters the relevant URLs, presenting only the relevant ones. In our system, although the user may manually give some information, the whole process can be automatic.

2 The architecture

The proposed architecture automatically retrieves relevant documents over the internet. It makes use of context information as input to perform the search and to filter the returned URLs. The mechanism is called *Metasearcher*.

IR019235

BEST COPY AVAILABLE

The document retrieval mechanism is implemented using an information agent. The architecture is made up of three modules: context reconnaissance module, search module and filter module, as we describe in the following sections.

3 Information agents

Information agents are programs that model an information space of a user. They are not well defined. An agent are characterised for operating at high levels of abstraction and usually for the use of distributed resources. Like specialist systems, "information systems are hard to be characterised but easy to be identified" [Cybenko et. al 1990]. However, unlike a specialist system that models one specialist (one person) and makes his knowledge available to many users, an agent models one user, his needs of information and actions. Therefore, an agent must be customised for each user, making programmability a basic requirement for it. A definition of an information agent can be found in [Cybenko et. al 1990].

3.1 The architecture of the information agent

The proposed information agent models a human assistant that creates links using a behaviour similar to a human being [ACM, 1997]. A human assistant would read the information searched by an user and would extract the context from this information. In the next step, he would define keywords to search the internet using existing indexing mechanisms (e.g. Altavista). Finally, he would "read" the returned documents and select those that are really relevant in the context identified. Finally, he would build an HTML page to be presented.

To perform its role, the agent executes three basic tasks: first it reads the context information and identify keywords; second it submits searches to indexing mechanisms and finally it filters the information returned using the context obtained in the first task as model. The process is shown in Figure 1 where:

- The context reconnaissance module identifies the context and creates keywords to be presented to the search module. The scheme used to generate the keywords is shown in Figure 2. This module also creates a semantic space (using LSI [6]) from the context information.
- The semantic space will be used in the filter module to verify relevance of returned documents. Context information can be obtained from Internet browser caches, users' bookmarks, a set of documents regarded as interesting, etc.
- The search module submits (in parallel) the actual search to indexing mechanisms, receive the responses and assembles an HTML page to be submitted to the filter module.
- Finally, the filter module retrieves the full documents whose URLs were returned in the search module. It then retrieves (in parallel) the full documents and adds them to the semantic space computed in the first module. Documents that position themselves near the existing documents are regard relevant and presented to the final user. Documents that do not stay near the existing ones are discarded. The filtering module is the kernel of the proposed architecture is explained in detail below.

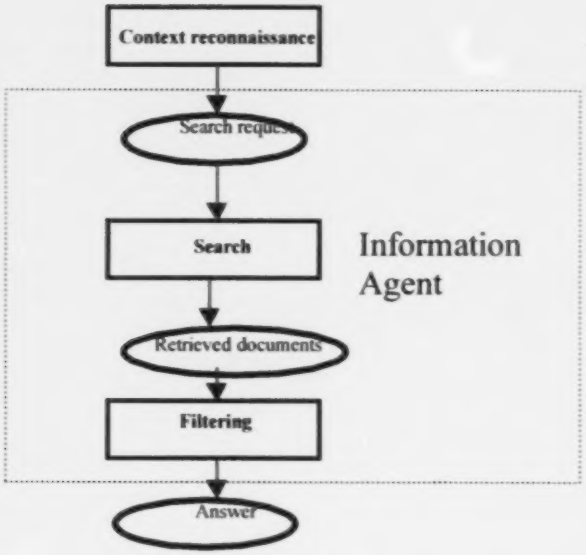

Figure 1: The Architecture of the Agent

3.2 The filter module

The search using keywords has many drawbacks. The search may fail because of synonyms and polysemia (more that one meaning to the same word). To reduce these problems several filtering techniques can be used [Foltz & Dumais, 1992]. We have decided to use LSI [Dumais, 1991] [Dumais et al. 1988] to filter the documents retrieved.

LSI takes advantage of the higher implicit order of the association of terms to documents in order to create a multi-dimensional semantic structure of the information. Using the patterns of co-occurrence of words, LSI is able to infer the structure of relationship between terms and documents. The singular Value Decomposition of the association term-document matrix is obtained producing a matrix, with reduced dimensions, with the k best orthogonal factors to approximate the original matrix to the "semantic space" model for the collection. This semantic space reflects the main associative patterns in the data ignoring some minor variations that may be produced by idiosyncrasies in the use of the term in particular documents. Therefore, LSI produces a representation of the adjacent information "latent" semantics.

Because LSI produces an adjacent similarity semantics space, documents on similar topics tend to be grouped in the space. That is the base of the use of LSI. We first create a space based on information known to be (or regarded as) relevant and then new documents are added to the space. If they position themselves near the existing ones, they are regarded as relevant, otherwise they are discarded.

To implement LSI, a term-document matrix is built. The elements in the matrix are the occurrences of each term in a document. Therefore, the matrix can be represented by $A = [a_{ij}]$ where a_{ij} denotes the frequency that term i occurs in document j. Global and local weights can be applied [Dumais et al. 1988] to increase or decrease the importance of terms within between documents. We can then write $a_{ij} = L(i,j) \times G(i)$ where L(i,j) is the local weight of term i in document j, and G(i) is the global weight of term i. Matrix A is factored in a product of three matrixes using the Singular Matrix Decomposition (SVD): $A = U \ \Sigma V^T$.

SVD derives the model of the latent semantic structure from the orthogonal matrixes U and V with the left and right singular vectors of A respectively, and the diagonal matrix Σ, of the singular values of the original relationships into linear independent vectors. The use of K factors is equivalent to approximate the original term-document matrix by A_k as defined in equation

$$A_K = \sum_{i=1}^{K} u_i \cdot \sigma_i \cdot v_i^T \text{ , where}$$

A_K = best approximation of rank k to matrix A	m = number of terms
U = term vectors	n = number of documents
Σ = singular values	k = number of factors
V = document vectors	r = Rank of A

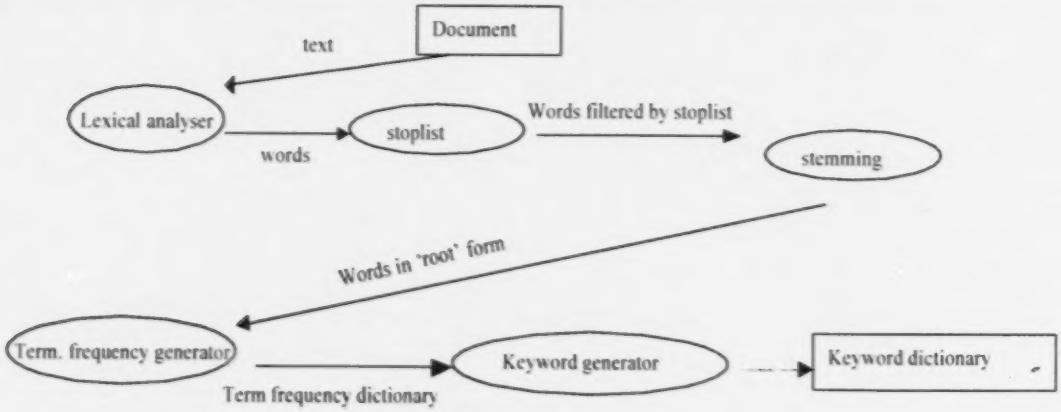

Figure 2: Keywords generation process

Figure 3 is a mathematical representation of the decomposition into singular value. U and V are the term and document vectors, and Σ represents the singular values. The shaded regions in U and V the diagonal line in Σ represents A_k.

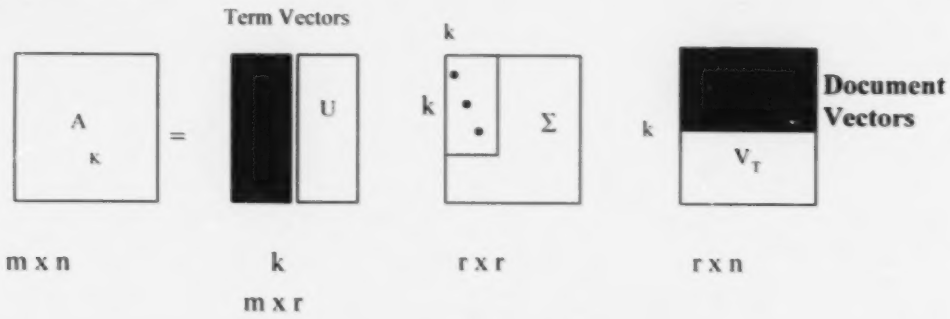

Figure 3: Decomposition into singular value

SVD captures most of the adjacent structure in the association terms-documents and, at the same time, it removes the variability in the use of words. Intuitively, as the number of dimensions k is much smaller than the number of unique terms m, minor differences in the terminology will be ignored. Terms that occur in similar documents, for example, are close even if they do not occur in the same document. That means that documents that do not share any words with the keywords in the search may be close to a keyword in the k-space. This representation captures term to term associations and it is used to retrieve information. The main idea in LSI is to model the inter-relationships among terms and use them to improve retrieval.

As we have explained, *Metasearcher* retrieves documents obtained from the indexing mechanisms. To be filtered, these documents are used in a "query" to obtain their similarity to relevant documents in the vetorial

BEST COPY AVAILABLE

space. One query is a set of words. The retrieved document, or query, must be represented in the k-dimension space as $\hat{q} = q^T U_k \Sigma_k^{-1}$, where q is the vector of words in the new document, that can be multiplied by the appropriate term weights. The sum of these k-dimension vectors is reflected by term $q^T U_k$ and the right multiplication by Σ_k^{-1} weights differentially individual dimensions. Thus, the query vector is located in the weighed sum of its constituents term vectors. The query vector can then be compared to all existing document vectors, using a similarity function. One similarity function is the cosine between the query vector and the closest vector in the vectorial space of relevant documents. Usually, if the cosine exceeds a threshold the retrieved document is regarded as relevant [Dumais, 1991].

Documents not regarded as relevant are discarded and relevant ones are presented to the output system as an HTML document.

4 Scenarios of usage

Metasearcher can be used together with several systems. It is implemented as a modular architecture that allows the input and output modules to be changed to adapt to different requirements.

For example, the input system can collect data from one user and use that information to retrieve other document on the same subject, or the input can come from an intranet supporting a co-operative environment as it has been tested in [Badue et. al, 1998]. In this case context information (including information manually entered), and returned documents are shared by a group. Another scenario would be the one where a student who needs to write an essay to "train" *Metasearcher* with a set of papers on the subject and have it to retrieve other related documents.

5 Performance

The system is still being tested. It has been used by a group of five students who are given a task and should search the Internet (starting with an empty cache) until they build a cache of about 1.5MB of html documents for each user. After the cache is loaded, the system is left to work, usually overnight because of slow connections.

Although the number of documents retrieved by indexing engines vary a lot, typically 60% of them are filtered by the engine. We have found that the input system, specially the interface, has to be improved to allow users to manually discard non relevant documents. In tests where non relevant documents were discarded from cache manually, the rate of filtered documents raised to near 80%.

Although the results are very promising we still need more tests to assess the system. We have found that the input system has to be improved to allow users to input feedback with less effort, and to share their "filtering" with other users. Users have had the "feeling" that the system retrieves more relevant information when caches are shared than when they use their caches only. However, we have not yet assessed how correct that "feeling" is.

6 Acknowledgements

This work was partially supported by CNPq/PROTEM-CC project SAM Sistemas de Autoria Multimídia.

7 Conclusion

This work has proposed an architecture based on information agents to automatically retrieve documents in the Internet and filter the returned documents, using LSI, according to context information. The proposed architecture makes use of indexing mechanisms and is modular, in the sense that its input and output modules can be changed.

To perform its role, the system obtains context information from a source that can be Internet browser caches, a directory of papers, etc. Then it uses that information to generate keywords that are submitted to existing indexing mechanisms, the returned URLs are retrieved an the full documents are matched to the context using LSI. Finally, it generates an HTML page that is passed on to the output system that may perform further computation or just present the page to the final user.

Metasearcher has been "plugged" to an intranet environment in an University environment, where it is used to support co-operative learning. Currently, the system is being tested and data is being gathered to asses its effectiveness as a learning aid.

8 References

[ACM, 97] Association for Computing Machinery. Intelligent Agents. *Communications of the ACM*, 37 (7), July 1994.

[Badue et al, 1998] BADUE, C. and W. VAZ. *An information agent and database environment to support collaborative work in the Internet (in Portuguese).* RT-INF-UFG 02/98. Universidade Federal de Goiás, Goiânia-GO, 1998.

[Cybenko et. Al, 1994] CYBENKO, G., R. GRAY, Y. WU and A. KHRABROV. *Information Architecture and Agents*. Thayer School of Engineering, Dartmouth College. Hanover,1994.

[Deerwester et. Al. 1990] DEERWESTER, S., S. DUMAIS, G. FURNAS, T. LANDAUER and R. HARSMAN. 1990. *Indexing by latent semantic analysis,* Journal of the American Society for Information Science, 41, pp. 391-407.

[Dumais et. Al. 1988] DUMAIS, S. T., G. W. FURNAS, T. K. LANDAUER, S. DEERWESTER & R. HARSHMAN. Using latent semantic analysis to improve access to textual information. *Proceedings of the Conference on Human Factors in Computing Systems*, CHI. 281-286, 1988.

[Dumais, 1991] DUMAIS, S. T. Improving the retrieval of information from external sources, Behaviour Research Methods, Instruments, & Computers, 23, pp. 229-236, 1991.

[Foltz & Dumais, 1992] FOLTZ, P. W. e S. T. DUMAIS. Personalised information delivery: An analysis of information filtering methods, *Communications of the ACM* 35, 51-60,1992